

REMARKS

Pending Claims:

In this application, claims 1-6 and 15-20 are currently pending. Claims 1, 15, and 18 have been amended, while the remaining claims have not been altered in this Response.

Rejection under 35 U.S.C. §101:

The applicant has amended independent claims 1 and 15 to affirmatively state that the data cells are stored on a computer system. These amendments should overcome the rejection under 35 U.S.C. §101.

Rejection under 35 U.S.C. §102(b)

The Examiner has rejected claims 1-6 as anticipated by Berner, U.S. Patent No. 5,907,846, while claims 15-20 were rejected as obvious over Berner in view of Cheng, U.S. Patent No. 6,356,896. The office action considered Figure 3 of Berner to show the cells of the present invention. Cheng is used to cite the existence of one table sharing columns with two other tables.

The Applicant notes the similarity between the present office action and the office action of July 23, 2003, which cited the Kawai patent (U.S. Patent No. 5,717,924) and the Williamson patent (U.S. Patent No. 6,122,641). As in the current office action, these references did not discuss the use of cells containing the specific requirements set forth in claim 1. Instead, the office action appeared to represent a belief that the then pending claims read on any relational database tables having multiple field values. On January 19, 2004, the Applicant responded to the July 23, 2003 office action with a detailed, fifteen-page Response. In an office action dated March 4, 2004, the Examiner allowed claims 1-6 and 15-20. The Applicant then canceled the remaining claims leaving only the allowed claims, anticipating a Notice of Allowance. Instead, the current office action was received.

Cells (Claims 1-6)

In response to the current office action, the Applicant has amended claim 1, further clarifying the scope of the present invention. As amended, claim 1 covers a collection of data comprising "a plurality of data cells," with each cell being a data

construct. Like other data constructs used for the storage of data, the cell data construct of claim 1 contains an attribute value associated with an attribute type. Unlike other data constructs, each cell of newly amended claim 1 must contain “the attribute value for *only the single attribute type* and for *only the specific instance of the specific entity type*.” To further clarify this, claim 1 now specifically states that the cell “does not contain the attribute value for any other attribute type or any other instance of the specific entity type.” Thus, in order for a reference to anticipate claim 1, the reference must teach a data construct that contains the attribute value for only the single attribute type and for only the specific instance of the specific entity type and no other attribute value.

In addition to the attribute value, claim 1 requires that the cell data structure contain within itself a single instance identifier and a single attribute type identifier. Claim 2 adds that the cell must also contain an entity identifier value. It is these four elements (instance identifier, attribute type identifier, attribute value, and entity identifier) that allow each cell to be “self-identifying,” which is an additional limitation of newly amended claim 1. The self-identifying nature of each cell is vital for the functioning of a data collection made up of data cells, as is explained in the Specification:

Because there is no external construct to associate one cell 100 with another, each data cell 100 of the present invention must be self-identifying. In other words, the data cell 100 must contain not only the value of interest, but it also must contain enough information to identify the attribute to which the value relates, and to associate the attribute with a particular instance of an entity.

Specification, page 7, lines 12-19.

None of the cited prior references discloses or suggests a self-identifying data construct that contains the specific limitations of claim 1. Figure 3 of Berner teaches three logical views to the Employee and Project class shown in Figure 2, with each logical view being represented as a table with four fields. Berner, col. 6, line 53 to col. 7, line 21. Technically, these tables 302-306 are not separate data structures, but are merely “logical views” of the object classes shown in Figure 2. Berner, col. 7, lines 17-21. Nonetheless, the Applicant acknowledges that the creation of the tables shown in Figure 3 as actual data constructs is well within the scope of the prior art.

The office action argues that tables 302-306 of Berner show a plurality of cells as required in claim 1 of the pending application. However, while these tables are data constructs that contain attribute values, these tables do not meet the specific limitations

of claim 1. Specifically, the tables do not contain “*only* the one specific attribute type and for *only* the one specific instance of the specific entity type. The tables of Berner, like all database tables, contain multiple rows with each row containing attribute values for a different instance of an entity. For instance, the first row in table 302 contains attribute values for Employee 111, while the second row contains attribute values for Employee 222. Inside of each row are multiple fields of data, with each field containing a separate attribute value (i.e., the ID, name, age, and gender of a specific employee). Thus, the tables 302-306 are no different than any prior art data table, and contain multiple attribute values for multiple instances of an entity. Thus, the tables cannot be considered a data construct containing an attribute value for only the single attribute type and for only the specific instance of the specific entity type. The clarification of claim 1 that the cell “does not contain the attribute value for any other attribute type or any other instance of the specific entity type” makes it even clearer that a database table such as those of Berner does not constitute a cell as defined by claim 1.

Although not stated as such in the office action, one might consider whether an individual row in a data table (or even the content of an individual field inside of a row) might constitute a cell as defined by claim 1. Given the language of the amended claim 1, it is clear that this is not the case. First, a single row in a prior art table is not considered a separate data construct, but merely a portion of a table construct. This is clearly the case when one realizes that a row cannot stand on its own. It is relevant only within the definition of a table, which indicates the type of information that is found within each field through its column definitions. Second, a typical row in a database table will contain multiple attribute values for a single entity. This second distinction, however, may not apply to single column database table, in which each row has only a single attribute value. In this case, the row is effectively a single field value, such as “Jane D.” This then becomes mere data, not a data construct.

But even if one assumes an individual row or field in a table is a separate data construct, prior art database rows and fields do not meet the other requirements of the claims. Specifically, claim 1 requires that each cell have a single instance identifier, a single attribute identifier, and an attribute value. Claim 2 adds the requirement for an entity identifier value in that cell. To avoid having the definition of a cell in claims 1 and 2 cover any four element data construct, the claims have been carefully constructed to define these four elements as follows:

a single instance identifier value	identifying one specific instance of a specific entity type
a single attribute type identifier value	identifying one specific attribute type for the specific entity type
an attribute value	for the identified one specific attribute type
a single entity identifier value	identifying the specific entity type

A single field within a row (or a one-column table row) certainly does not have all of these elements. Furthermore, rows in a prior art database table, such as the rows in Berner's tables 302-306, do not contain these elements. The office action states that the instance identifier is "ID 111," the attribute type is "job grade K5 (L5)," the entity identifier value is "ID 111," while the attribute value is not specifically identified. The office action also states that the specific entity type is "name Jane." To the best of the Applicant's understanding, the office action is mixing row values with field names. If one looks only at the values within a row of a table, which one must do if one considers the row itself to be the cell data construct, the office action has assigned "111" as the instance identifier value, "K5" as the attribute type, and either "111" or "Jane" as the entity identifier value. This is not consistent with the claim limitations found in claims 1 and 2, as the instance identifier must identify an instance of the entity type, which would mean that "111" defines an instance of type "Jane." This is clearly not correct. In addition, the attribute value must be a value for an attribute type, meaning that the "35000" value in the row is a value for attribute type "K5," which is also inaccurate. There is no variation in how the fields in the database tables of Berner are mapped to the claim elements of claims 1 and 2 that will allow the Berner table to anticipate claim 1. The attribute type identifier must appear within the same data construct as the attribute value. The entity type identifier must also appear with the same data construct as the instance identifier. If a single field or row of a table is compared to this definition of a cell, it will not have these elements. If the entire table is compared to this definition, the table will be found not to meet the requirement that each cell contain the attribute value for only the specific attribute type and only the specific instance of the specific entity type.

Furthermore, a standard row in a table will not meet the claim requirement that each cell be self-identifying. In a prior art databases, data identifying information is built into the construct of a table or the construct of the object that contains the data. Outside of the context, the data itself means nothing:

111	Jane D.	L5	35000
-----	---------	----	-------

This is just random data stuck together, with no indication as to the meaning of the numbers "111" or "35000." When this information is put into the context of a predefined table, however, the data has value:

ID	Name	Job Grade	Salary
111	Jane D.	L5	35000

In contrast, when this information is put into the self-defining data cells of the present invention, it exists as pure data outside of any external construct and yet maintains its value:

Employee	111	Name	Jane D.
Employee	111	Job Grade	L5
Employee	111	Salary	35000

This is because every data cell defines not only an attribute value ("Jane D."), but also the attribute type (Name) associated with that attribute value. Similarly, the cell of the present invention defines not only the instance identifier (111), but also the entity of which this cell is an instance (Employee). Once a cell data construct contains not only the attribute value, but the attribute type and the instance identifier, it becomes self-identifying. This self-identifying nature of each cell is an intricate part of the present invention. This has been added as an element of claim 1, and is not taught or suggested in the prior art.

The office action also states that the limitation in claim 3, namely identifying a cell set as a group of cells having the same instance identifier value and the same entity identifier value, is shown by the fact that each Berner table 302, 304, and 306 contain an row with a name value of Jane and an ID value of 111. If this were to meet the limitation of claim 3, then one of these two values must be the entity identifier identifying the specific entity type. Assuming that 111 is the instance identifier, the value Jane would

have to identify the type of entity for the cells (meaning that this cell relates to instance 111 of all Jane-type entities). However, Jane is clearly an attribute value for a particular attribute type (name), and the entity type is an Employee or Person. Under the language of the claims, the entity identifier must identify the entity type and must be repeated in the cells along with the instance identifier in order to define a cell set. This is not shown in Berner.

Associating Data Cells (Claims 15-20)

Newly amended claim 15 defines a collection of data cells, with a first and second data cell each having the same format and fields. The first and second cells both contain a first field, a second field, a third field, and a fourth field. Claim 15 also requires a linking cell that defines an association between the first and second data cells. The linking cell is defined as having two of its four values being the same as the first and second field of the first cell, with the remaining two values of the linking cells being the same as first and second field of the second data cell. To anticipate newly amended claim 15, a prior art reference must therefore teach a linking cell “defining an association between the first cell and the second cell” by combining the same two fields from the first and second cell into the four fields of the linking cell. Finally, claim 15 requires that each cell “contains a single element of data relating to a specific instance of an entity.”

These elements are not found in the prior art. Berner shows only rows in different tables containing the same the ID and name field. There is no third, linking cell defining an association between a first and second cell. Nor does any row in the Berner tables contain two field values from each of two rows. The cited table in Figure 4 of Cheng likewise does not show the invention of claim 15. This table 400 does not contain two columns from two separate table, but rather contains only the columns from the employee table 200. Cheng, col. 6, lines 9-13. However, table 300 in Figure 3 does contain columns from two tables joined together. Cheng, col. 5, lines 44-50. Nonetheless, this joined table does not teach the elements of claim 15. In particular, assuming a row of table 300 to be the linking cell, this row must have the same format and the same fields of the first and second cells. It is clear that this is not the case, as the columns of table 300 do not match the columns of any of the tables found in Figure 2 of Cheng. Furthermore, claim 15 requires that the first and second cell have the same formats and

the same fields, and that the values of the four fields in the linking cell be the same as field 1 and field 2 of the first cell, and field 1 and field 2 of the second cell. This is not found in Cheng, Berner, or any of the cited prior art references.

Claims 16 through 19 depend from claim 15, and further define the format of the cells. Claim 17 further includes a requirement that the linking cell utilize a flag to indicate that the cell contains linking information, a limitation that is not found in the cited prior art. The cited section in Berner relates only to an "on-cache status data field" that indicates whether a specific object is currently available on the cache. Berner, col. 8, lines 40-41. Such a field is unrelated to a flag that indicates whether a particular cell contains linking information. Claim 18 defines the four fields of the cells as being an entity instance field, an entity type field, and attribute type field, and an attribute value field. This is not found in the cited prior art, as explained in more detail above in connection with claims 1 and 2. Finally, claim 19 defines which two field types of the first cell and which two field types of the second cell make up the four field types in the linking cell. The office action cites Cheng as teaching this limitation because the result table 400 contains fields from two other tables. The applicant first notes that the Examiner is incorrect in its reading of Cheng, as explained above. Furthermore, even if this were an accurate reading, it is not relevant to the claim restrictions of claim 19. This claim requires that all three cells contain an entity instance field, an entity type field, an attribute type field, and an attribute value field, and that particular fields of the two linked cells are found in specific fields of the linking cells.

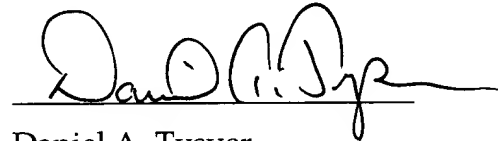
Finally, claim 20 depends from claim 19, and requires the creation of a second linking cell. The office action asserts that this is taught in Figure 3 of Berner. However, this figure does not show a single linking cell, let alone two complementary linking cells as specifically set forth in claim 20. Thus, for the same reasons described above, the applicant requests the withdrawal of this rejection.

CONCLUSION

All of the claims remaining in this application should now be seen to be in condition for allowance. The prompt issuance of a notice to that effect is solicited.

Respectfully submitted,
INFOBIONICS
By its attorneys:

Date: 6 May 2005

A handwritten signature in black ink, appearing to read "Daniel A. Tysver", written over a horizontal line.

Daniel A. Tysver
Registration No. 35,726
Beck & Tysver, P.L.L.C.
2900 Thomas Ave., #100
Minneapolis, MN 55416
Telephone: (612) 915-9634
Fax: (612) 915-9637